Equivalence Between Learning in Perceptrons with Noisy Examples and Tree Committee Machines

Mauro Copelli*, **Osame Kinouchi**† and **Nestor Caticha**‡
Instituto de Física, Universidade de São Paulo
Caixa Postal 66318, 05389-970 São Paulo, SP, Brazil

Abstract

We study learning from single presentation of examples (*incremental* or *on-line* learning) in single-layer perceptrons and tree committee machines (TCMs). Lower bounds for the perceptron generalization error as a function of the noise level ϵ in the teacher output are calculated. We find that optimal local learning in a TCM with K hidden units is simply related to optimal learning in a simple perceptron with a corresponding noise level $\epsilon(K)$. For large number of examples and finite K the generalization error decays as α_{cm}^{-1}, where α_{cm} is the number of examples per adjustable weight in the TCM. We also show that on-line learning is possible even in the $K \to \infty$ limit, but with the generalization error decaying as $\alpha_{cm}^{-1/2}$. The simple Hebb rule can also be applied to the TCM, but now the error decays as $\alpha_{cm}^{-1/2}$ for finite K and $\alpha_{cm}^{-1/4}$ for $K \to \infty$. Exponential decay of the generalization error in both the perceptron learning from noisy examples and in the TCM is obtained by using the learning by queries strategy.

Key words: *rule extraction, generalization, learning algorithms, on-line learning, perceptron, multi-layer networks, committee machine.* **Short title:** On-line Learning in Committee Machines.

PACS. #: 02.50, 05.90, 64.60 Cn, 75.10 Hk, 87.10.e+10, 87.70.+c.

*E-mail: copelli@if.usp.br
†E-mail: osame@if.usp.br
‡E-mail: nestor@if.usp.br

1 Introduction

Learning regularities present in the environment is a central task for organisms. Indeed we can think the sensory and memory systems as auxiliary devices to the primordial function of extracting, from the external and internal media, useful rules for appropriate responses. In the studies of artificial neural networks the task of extraction of underlying and perhaps non apparent rules from raw examples (generalization) has been also recognized as a central issue, displacing the previous emphasis on associative memory properties of such networks.

The simplest neural network architecture with generalization properties is the single-layer perceptron. This network has been studied with great detail by the statistical mechanics community and provides a test ground for new ideas on learning algorithms and strategies [1].

One of these strategies recently studied is learning from single presentation of examples (often called *on-line*, *incremental* or *sequential* learning [2, 3, 4]) where the examples are used sequentially inducing a single change in the network (being discarded after that). Besides its biological appeal, it is a computationally very cheap procedure since no memory space is needed for storing the old examples, nor time expensive retraining is required. On-line learning is also the natural scenario for changing environments where old examples may no longer be representative of the actual rule [4, 5] and for selection of examples (*learning by queries*) strategies where each new example is chosen sequentially depending on the present state of the network [2, 6, 1].

Surprisingly, it leads to generalization performances comparable to those obtained by the conventional minimization of global cost functions defined over all examples (*off-line* learning) [6]. The on-line generalization error for randomly chosen examples decays asymptotically with the same power of $\alpha_p = P/N_p$ (the number of examples P per adjustable parameters N_p of the perceptron) as for off-line learning, the lower bound for this error being only twice the off-line Bayesian bound $e_g^{(p)}(\alpha_p \to \infty) \approx 0.442/\alpha_p$ in the perceptron [7].

Off-line learning has been extensively studied with methods of equilibrium statistical mechanics, mainly through Gardner coupling space analysis based on the replica formalism [1]. It is interesting to compare these results with those for on-line learning, specially for the case of multi-layer nets where the statistical mechanics calculations are more involved.

Indeed, on-line learning has been extended recently to multi-layer nets [8, 9, 4, 10, 11, 12]. Here we are interested in the determination of the *optimal performance* achievable by on-line learning when applied to a committee machine. The optimal generalization performance in a tree committee machine trained on-line with a *non-local* algorithm has been studied in [9]. Here we introduce an optimal *local* algorithm and we show that this problem is nicely related to the case of a single-layer perceptron learning from examples with noisy data.

The paper is organized as follows. In section 2 the on-line learning scenario is described and a general prescription for calculating on-line optimal performances is given. The optimal performance for a perceptron learning from corrupted examples is determined in section 3, as well as the performance of the simple Hebb algorithm. Section 4 contains the analysis of the optimal local learning algorithm for a tree committee machine with K hidden units; the relationship between the two problems is discussed in this section. The strategy of learning by queries in both problems is considered in section 5. Our conclusions are summarized in section 6.

2 On-line Learning in Neural Networks

2.1 Definitions

We study *supervised learning* of an unknown mapping $M : \mathcal{I} \to \mathcal{O}$ of a N_p-dimensional input space $\mathcal{I}$ into single output space $\mathcal{O}$ by using only the information from a set of input-output examples. The *training set* $\mathcal{L} = \{(\mathbf{S}^\mu, \xi^\mu)\}(\mu = 1, \ldots, P)$ is an ordered list of P examples, i.e. pairs of input vectors $\mathbf{S}^\mu = \{S_1^\mu, \ldots, S_{N_p}^\mu\}$ with the corresponding output signals ξ^μ. Average over the training examples will be referred to as an integration over a measure $d\nu_{\mathcal{L}}$.

For simplicity we will describe the formalism for a single perceptron which, whenever necessary, must be regarded to be a branch perceptron of a non-overlapping or tree committee machine (TCM). Quantities labeled by a superscript $^{(p)}$ refer specifically to the perceptron, while the results for the committee machine are denoted by $^{(cm)}$. The examples used by the perceptron with adaptive weights $\mathbf{J} \in \mathbb{R}^{N_p}$ (often called the *student* or *hypothesis*) are generated by another network with the same architecture but unknown weights $\mathbf{B} \in \mathbb{R}^{N_p}$ (the *teacher* or *rule*). This corresponds to a *realizable task*, which means that the rule to be inferred lies within the hypothesis space so that some proximity measure in this space can be defined. As usual the relevant measure is the average *rule-hypothesis overlap*

$$\rho = \left\langle \left\langle \frac{\mathbf{J} \cdot \mathbf{B}}{J\, B} \right\rangle_{\mathcal{N}} \right\rangle_{\mathcal{L}} , \tag{1}$$

where $J = |\mathbf{J}| = \sqrt{\mathbf{J} \cdot \mathbf{J}}$, $B = |\mathbf{B}| = \sqrt{\mathbf{B} \cdot \mathbf{B}}$, $\langle \cdots \rangle_{\mathcal{N}}$ is the average over the distribution of networks produced by the algorithm and $\langle \cdots \rangle_{\mathcal{L}}$ is the average over the possible training sets. It is possible to show that this quantity is self averaging in the thermodynamic limit $N_p \to \infty$.

Two general classes of behaviour appear, depending on whether the transfer function of the neurons is invertible or not. In the former case we have, for example, the linear neuron $g(x) = x$ and the graded response neuron $g(x) = \tanh(x)$. In the last case we have the Boolean neuron $g(x) = \mathrm{sign}(x)$ where the binary output conveys only partial information about the neuron local field, leading to a qualitatively different learning behaviour. On-line learning has been studied both in the invertible [4, 10, 12] as in the non-invertible [3, 6, 9, 13] cases. Although only machines with Boolean units will be studied in this work, we will show how the optimal performance can be obtained from a common prescription for both type of units.

The local post-synaptic fields in the teacher and student nets are defined as

$$b_\mu \equiv \mathbf{B} \cdot \mathbf{S}^\mu / B \ , \qquad h_\mu \equiv \mathbf{J} \cdot \mathbf{S}^\mu / J \ , \tag{2}$$

respectively. From the noise free rule response $\sigma_B^\mu \equiv \mathrm{sign}(b_\mu)$, the output data ξ^μ is generated using a conditional distribution $P(\xi \mid \sigma_B)$. To describe this situation we write

$$P(\xi \mid \sigma_B) = \left(1 - \frac{\epsilon}{2}\right) \delta(\xi, \sigma_B) + \frac{\epsilon}{2}\, \delta(\xi, -\sigma_B)\,, \tag{3}$$

where δ is the Kronecker delta. The quantity ϵ (twice the probability of flipping the noise free response σ_B) will be referred to as the noise level in the teacher output.

2.2 Performance measures

Three different relevant performance measures can be defined. The *training* or *classification* error measures the probability of a misclassification over the training set,

$$e_c^{(p)} \equiv \frac{1}{\alpha N} \sum_{\mu=1}^{\alpha N} \Theta\left(-\xi^\mu \sigma_J^\mu\right) \tag{4}$$

Contrasting with off-line learning, on-line algorithms always produce networks with nonzero classification error. This represents no problem concerning the generalization properties and perhaps constitutes an advantage in the case of noisy examples by avoiding overfitting. We observe that this quantity is measurable from simulations but up to now there is no analytical results for the on-line classification except for the simple Hebb rule [14].

The second measure is the *generalization* error, which measures the probability of misclassification of a *noise free* example chosen with a uniform test distribution $d\nu_T(\mathbf{S})$

$$e_g^{(p)} \equiv \int d\nu_T(\mathbf{S})\, \Theta\left(-\sigma_B(\mathbf{S})\sigma_J(\mathbf{S})\right) = \frac{1}{\pi} \arccos \rho\,, \tag{5}$$

Although this error is perhaps non accessible (if we only can obtain noisy outputs ξ as test examples) it has theoretical importance since it measures how close the hypothesis has been successful in approximating the rule.

A third measure, which we will call *prediction error*, is the probability of misclassification of a noisy example drawn from the same distribution used during learning,

$$e_p^{(p)} \equiv \int d\nu_{\mathcal{L}}(\mathbf{S}, \xi)\, \Theta\left(-\xi \sigma_J(\mathbf{S})\right)\ . \tag{6}$$

If this distribution is i.i.d. with $P(\xi \mid \sigma_B)$ given by eq.(3), we have

$$e_p^{(p)} = (1-\epsilon)\, \frac{1}{\pi} \arccos \rho + \frac{\epsilon}{2} = (1-\epsilon)\, e_g^{(p)} + \frac{\epsilon}{2}. \tag{7}$$

Thus, in the presence of noise the prediction error is non zero even if we have a perfect match $\mathbf{J} = \mathbf{B}$. This prediction error is measurable but, since ϵ is unknow, it does not give a direct estimate of the success of the modeling task (how near $\mathbf{J}$ is to $\mathbf{B}$). We observe, however, that both $e_g^{(p)}$ and $e_p^{(p)}$ are monotonic decreasing functions of the overlap ρ, and that the maximization of this overlap is the primordial task for the learning procedure.

2.3 Learning dynamics and the optimization procedure

We will show that tight upper bounds for the average evolution of ρ as a function of the number of examples and noise level can be easily calculated for any distribution of examples.

Our starting point is a fairly general learning dynamics which depends on the two vectors available in the problem (**J** and **S**)

$$J_i(\mu) = \left(1 - \frac{\Omega(\mu)}{N_p}\right) \; J_i(\mu-1) + \frac{1}{N_p} F(\mu) S_i^\mu \;, \tag{8}$$

where

- $F(\mu)$ is a function which weights the change induced in the synaptic vector **J** by a new example. Due to obvious motivations, we will call it the perceptron *modulation function.*
- $\Omega(\mu)$ is a decay parameter which may be used to control the length of vector **J**.

From eq. (8) it is possible to obtain in the limit $N_p \to \infty$ differential equations [2, 6, 5] describing the evolution of ρ and J as a function of the "continuous time" $\alpha_p = \mu/N_p$. These learning equations can be used for the calculation of any algorithm:

$$\frac{d\rho}{d\alpha_p} = \rho \int d\nu_{\mathcal{L}} \, \frac{F(\mu)}{J} \left(\frac{b(\mu)}{\rho} - h(\mu) - \frac{F(\mu)}{2J} \right) \;, \tag{9}$$

$$\frac{dJ}{d\alpha_p} = J \int d\nu_{\mathcal{L}} \left(\frac{F^2(\mu)}{2J^2} + \frac{F(\mu)h(\mu)}{J} - \Omega(\mu) \right) \;, \tag{10}$$

Defining F^{opt} as the weight function that optimizes $d\rho/d\alpha_p$, a simple variational calculation leads to

$$F^{opt}(\mu) = \frac{J}{\rho} \left(\langle b \rangle_{b|\xi,h} - \rho h \right) \;. \tag{11}$$

where $\langle \cdots \rangle_{b|\xi,h}$ stands for the average over $P(b_\mu \mid \xi, h_\mu)$ Note that, although in this paper we will restrict ourselves to the case where the examples are random i.i.d. variables distributed uniformly the above prescription for the optimal weight function is valid for *any* distribution of examples. The only hypothesis is that learning occurs obeying the initial dynamics (8).

Defining $\tilde{W} \equiv F^{opt}/J$, the evolution equations for the optimal algorithm become

$$\frac{d\rho}{d\alpha_p} = \frac{\rho}{2} \left\langle \tilde{W}^2 \right\rangle_{h,\xi} \;, \tag{12}$$

$$\frac{dJ}{d\alpha_p} = J \left[\frac{1}{2} \left\langle \tilde{W}^2 \right\rangle_{h,\xi} - \left\langle h^2 \right\rangle_h + \frac{1}{\rho} \langle hb \rangle_{h,b} - \Omega \right] \;. \tag{13}$$

Again, no assumptions on the distribution of examples $P(b,h)$ where made. Eqs. (12) and (13) are thus also valid for *any* distribution of examples. It follows that, whenever the distribution of examples satisfies the condition

$$\rho \left\langle h^2 \right\rangle_h = \langle hb \rangle_{h,b} \;, \tag{14}$$

and learning is done unconstrained ($\Omega = 0$), the evolutions of ρ and J are simply related,

$$\frac{1}{\rho}\frac{d\rho}{d\alpha_p} = \frac{1}{J}\frac{dJ}{d\alpha_p} , \tag{15}$$

as can be seen by inserting eq. (14) in eq. (13). This leads to the remarkable property

$$J(\alpha_p) = c \times \rho(\alpha_p) , \tag{16}$$

where c is a constant which can be self-consistently set equal to unity. In particular, eq. (14) holds for the uniform distribution of examples (see eq. (78) in Appendix A) and for the selected examples used in the "learning by queries" strategy. Result (16) will be shown to be very useful for the practical implementation of the optimal algorithm.

3 Perceptron Learning from Noisy Data

3.1 The optimal algorithm

In the Appendix we show that the optimal modulated Hebbian term can be easily calculated as

$$F^{opt} S_i^\mu = -J\lambda^2 \frac{\partial}{\partial \hat{J}_i} \ln P(\xi_\mu \mid h_\mu) , \tag{17}$$

where $\hat{J}_i \equiv J_i/|\mathbf{J}|$ and

$$\lambda \equiv \frac{\sqrt{1-\rho^2}}{\rho} . \tag{18}$$

This prescription for calculating the optimal modulation can be used both for boolean or graded response units.

We now give an explicit expression for the optimal weight function for the particular case where the examples are independent identical uniformly distributed random variables. Using

$$P(\xi \mid h) = \sum_{\sigma_B} P(\xi \mid \sigma_B, h) P(\sigma_B \mid h) = \sum_{\sigma_B} P(\xi \mid \sigma_B) P(\sigma_B \mid h) \tag{19}$$

and the result

$$P(\sigma_B \mid h) = H\left(\frac{-\sigma_B h}{\lambda}\right) , \tag{20}$$

(see Appendix) one easily obtains

$$P(\xi \mid h) = \frac{\epsilon}{2} + (1-\epsilon) H\left(\frac{-\xi h}{\lambda}\right) , \tag{21}$$

where

$$H(x) = \int_x^\infty Dt , \qquad Dt \equiv \frac{1}{\sqrt{2\pi}} e^{-t^2/2} dt .$$

The optimal weight function, following the prescription of eq. (17), is then

$$F^{opt}(\mu) = J\lambda \frac{(1-\epsilon)}{\sqrt{2\pi}} \frac{e^{-h_\mu^2/2\lambda^2}}{\left[\epsilon/2 + (1-\epsilon)H\left(-\xi_\mu h_\mu/\lambda\right)\right]}\, \xi_\mu \quad . \tag{22}$$

The optimization procedure has determined not only the form of the modulation function, but most importantly the variables upon which it depends. In particular note that the presence of ξ means that the learning algorithm amounts to a hebbian like term ($\xi\mathbf{S}$) modulated by a function of λ and h.

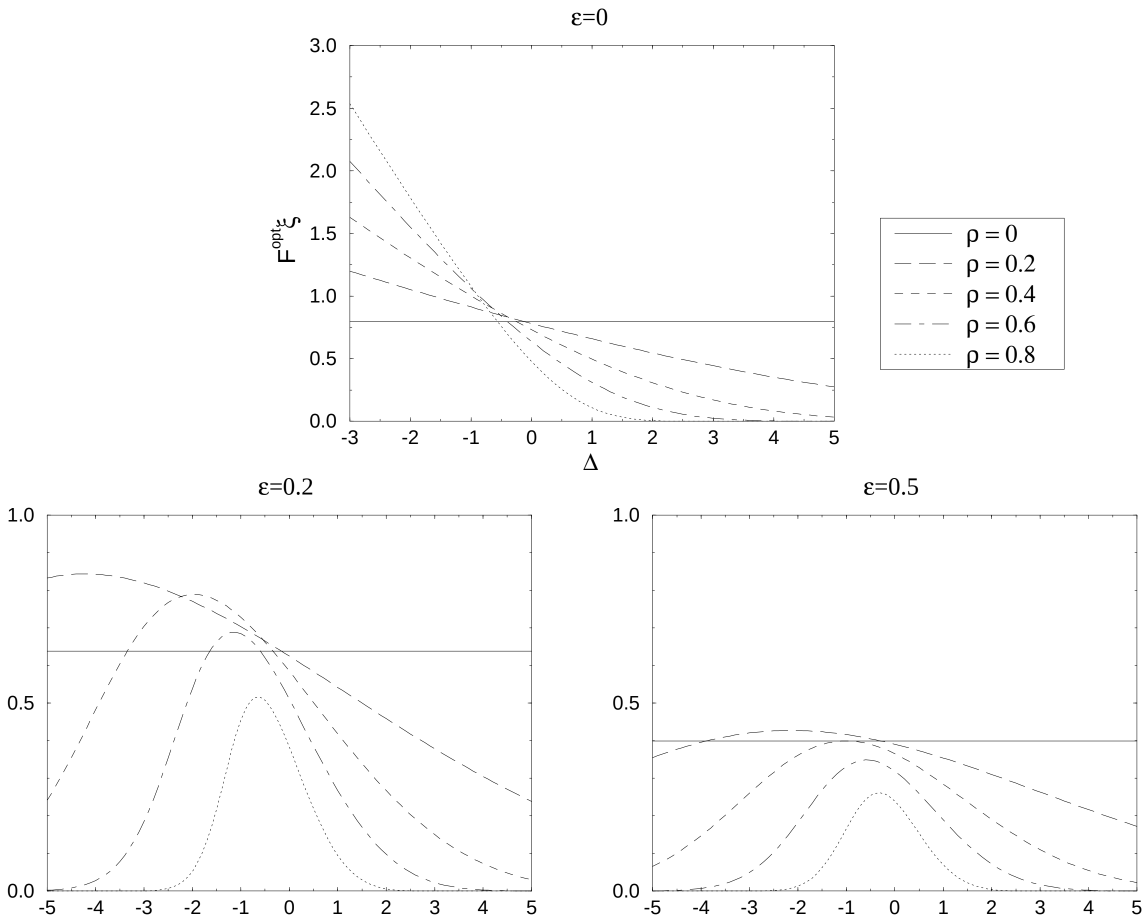

Figure 1: Norm of the modulation function ξF^{opt} against stability Δ for several values of ρ and ϵ. Note the qualitatively different behaviour for $\epsilon = 0$, which can also be observed in Figure 2.

The optimal weight function (22) presents some interesting properties (see Figure 1) which remind some biological aspects of memory modulation [15]. Its main characteristics are the dependence on

- the current performance, through the factor $\lambda = \sqrt{1-\rho^2}/\rho$;

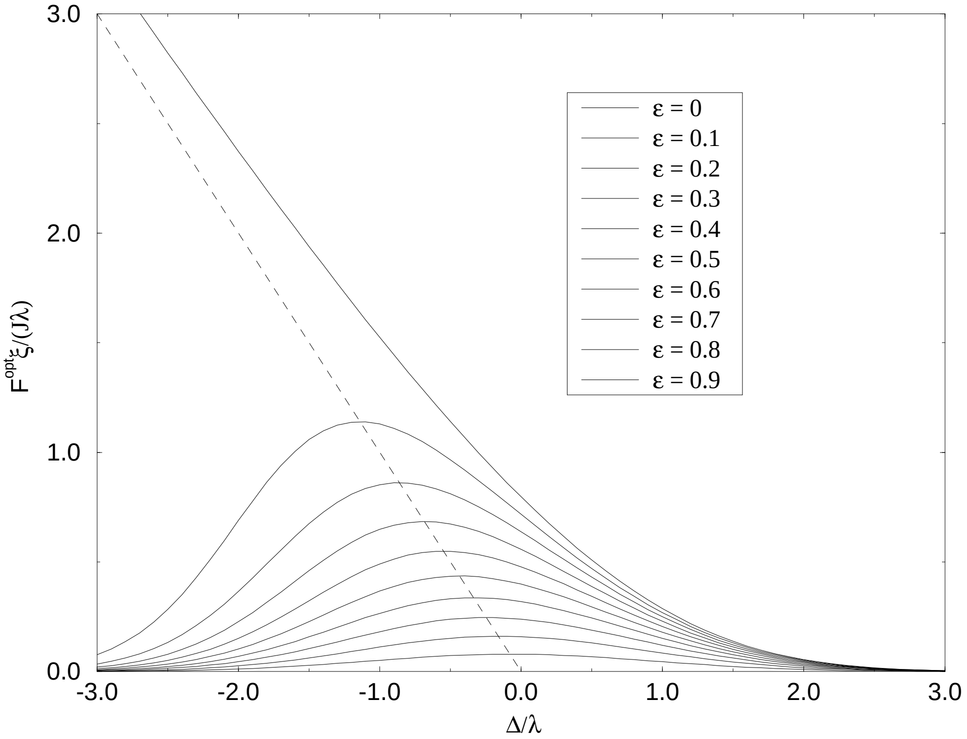

Figure 2: Rescaled weight function $\xi F/(J\lambda)$ against rescaled stability Δ/λ. From top to bottom, $\epsilon = 0, 0.1, \ldots, 0.9$. The dashed line corresponds to the maxima of the functions, and can be seen to satisfy $F^{opt}(y_0)/J\lambda = -y_0$, where $\Delta/\lambda = y_0$ is the solution of $d(F^{opt}/J\lambda)/d(\Delta/\lambda) = 0$.

- the "surprise" presented by a new example, as measured by the pre-training stability $\Delta_\mu \equiv \xi_\mu \mathbf{J}(\mu-1)\cdot \mathbf{S}^\mu/J$;
- the noise level ϵ present in the environment.

Thus, any practical algorithm which intends to approximate the theoretical optimal performance must approximate these characteristics of the optimal modulation function. In particular, the dependence in the unknown quantities λ and ϵ must be replaced by a dependence on measurable ones. This will be discussed at the end of this section, and in the meantime we will focus our attention on the interesting features presented by F^{opt}.

At $\rho \approx 0$ the weight function F^{opt} is a constant, *i.e.* the student perceptron starts learning in a pure Hebbian way, while at later times F^{opt} acquires a highly structured shape, as can be seen in Figure 1. The modulation is not static but performance dependent, varying along the learning process.

The dependence on the "surprise" content of a new example can be seen by considering the pre-training stability, defined by $\Delta \equiv h\xi$. When $\Delta > 0$, the student output equals the teacher noisy final output, *i.e.* the student is correctly predicting the example. In this situation the weight function is small, indicating that no major change in the synaptic couplings should be made. But for $\Delta < 0$ the student network is mispredicting the example, and thus the weight function strongly increases with increasing $|\Delta|$, *i.e.*, the surprise of being wrong makes the student attach importance to the example.

The presence of noise, however, changes this scenario. If Δ happens to be very negative, the weight function decreases, since, for such large $|h|$, the misprediction is probably due to noise in the teacher answer. This is a very nice result which agrees with the intuitive procedure of ignoring highly deviant data. The optimal modulation function suggests how to construct an algorithm robust to "outliers" and wrong data. The dependence on the "surprise" and noise level can be seen independently of the learning stage in Figure 2, where the rescaled weight function $F^{opt}\xi/J\lambda$ is plotted against the rescaled stability Δ/λ for several values of ϵ.

From eqs. (12) and (13) the evolution of ρ and J is obtained:

$$\frac{d\rho}{d\alpha_p} = \frac{(1-\rho^2)^{3/2}}{\rho^2}\frac{(1-\epsilon)^2}{2\pi}\int \frac{Dx\ e^{-x^2/2\rho^2}}{\epsilon/2+(1-\epsilon)H(x)} \tag{23}$$

$$\frac{dJ}{d\alpha_p} = J\ \frac{(1-\rho^2)^{3/2}}{\rho^3}\frac{(1-\epsilon)^2}{2\pi}\int \frac{Dx\ e^{-x^2/2\rho^2}}{\epsilon/2+(1-\epsilon)H(x)} - \Omega J\ . \tag{24}$$

Since the ρ equation is not coupled to the J equation, the evolution of $\rho(\alpha_p)$ is obtained by a simple numerical integration. The asymptotical behaviour for $\rho \to 1$ is described by

$$\frac{d\rho}{d\alpha_p} \simeq \frac{(1-\rho^2)^{3/2}}{\rho^2}\frac{I(\epsilon)}{2\pi}\ , \tag{25}$$

where

$$I(\epsilon) = (1-\epsilon)^2\int \frac{Dx\ e^{-x^2/2}}{\epsilon/2+(1-\epsilon)H(x)}\ . \tag{26}$$

A simple power counting on eq. (25) shows that

$$\rho(\alpha_p) \simeq 1 - \frac{2\pi^2}{I^2(\epsilon)}\frac{1}{\alpha_p^2}. \tag{27}$$

so that

$$e_g^{(p)} \simeq \frac{2}{I(\epsilon)}\,\alpha_p^{-1}\ , \tag{28}$$

for large α_p. The above equation shows that for any $\epsilon \neq 1$ asymptotical perfect generalization ($e_g^{(p)} \to 0$) is possible, and the error decays always with the same power law (α_p^{-1}). For $\epsilon \simeq 1$, $I(\epsilon)$ goes to zero quadratically,

$$I(\epsilon) \stackrel{\epsilon \to 1}{\simeq} \sqrt{2}(1-\epsilon)^2 \; . \tag{29}$$

The above result will be important for the analysis of the tree committee machine, in the next section.

We stress that these results represent lower bounds to the on-line performance of the perceptron. But in a real situation, one can hardly access the "noise level" of a system, while our optimal weight function explicitly makes use of ϵ. A noise estimator through which this problem is overcome has been proposed in [13] .

The optimal function also depends on the unaccessible overlap ρ. There are several ways out of this problem. A simple way for stationary rules is to take advantage of property $J(\alpha_p) = \rho(\alpha_p)$ (16) presented by unconstrained learning. The value of the perceptron norm J is a measurable quantity and the optimal function in this case reads

$$F^{opt} = \sqrt{1-J^2}\,\frac{(1-\epsilon)}{\sqrt{2\pi}}\frac{\exp\left(-\frac{1}{2}h^2J^2/(1-J^2)\right)}{\left[\epsilon/2 + (1-\epsilon)H\left(-\xi hJ/\sqrt{1-J^2}\right)\right]}\,\xi \quad . \tag{30}$$

The simulations presented in this paper have used this form for F^{opt}. In the case of time dependent rules these solutions do not work, but a method for on-line estimation of λ has been developed in [5].

3.2 The Hebb algorithm

The Hebb algorithm, initially studied by Vallet [14] for the perceptron, has recently been applied in the tree committee machine with three hidden units [9]. We now study the performance of the Hebbian algorithm in the case where the teacher outputs are corrupted by noise. This result, through the equivalence property to be presented later, enables a generalization of our results to TCMs with any K.

Instead of looking directly at the pure Hebb algorithm ($F = \xi$) we allow for a possibly varying function of the learning stage

$$F_H = W(\alpha_p)\xi \; . \tag{31}$$

Inserting this weight function in the evolution equation for ρ one obtains

$$\frac{d\rho}{d\alpha_p} = (1-\epsilon)\sqrt{\frac{2}{\pi}}\,(1-\rho^2)\,\frac{W(\alpha_p)}{J} - \rho\frac{W^2(\alpha_p)}{2J} \; . \tag{32}$$

The weight function W^* that optimizes $d\rho/d\alpha_p$ is then

$$W^* = J\sqrt{\frac{2}{\pi}}\frac{(1-\rho^2)}{\rho}\,(1-\epsilon) \; , \tag{33}$$

in which case the evolution of ρ can be easily seen to be given by

$$\frac{d\rho}{d\alpha_p} = \frac{1}{\pi} (1-\epsilon)^2 \frac{(1-\rho^2)^2}{\rho} \ . \tag{34}$$

For the initial condition $\rho(0) = 0$ we have then

$$\rho(\alpha_p) = \left[1 + \frac{\pi}{2(1-\epsilon)^2 \alpha_p}\right]^{-1/2} \ . \tag{35}$$

Note that again the optimization procedure left the ρ equation uncoupled from the J equation. It is very interesting that the obtained weight function (33) corresponds to the usual Hebbian algorithm ($F = \xi$). To see this first calculate the J evolution (10) for $W = W^*$:

$$\frac{dJ}{d\alpha_p} = J\left[\frac{1}{\pi}(1-\epsilon)^2 \frac{(1-\rho^2)^2}{\rho^2} + \frac{2}{\pi}(1-\epsilon)^2(1-\rho^2) - \Omega\right] \ . \tag{36}$$

From eq. (33) one obtains

$$\frac{dW^*}{d\alpha_p} = (1-\epsilon)\sqrt{\frac{2}{\pi}}\left[\frac{1-\rho^2}{\rho}\frac{dJ}{d\alpha_p} - \left(2 + \frac{1-\rho^2}{\rho^2}\right) J \frac{d\rho}{d\alpha_p}\right] \ . \tag{37}$$

Eqs. (34) and (36) lead then to

$$\frac{dW^*}{d\alpha_p} = -\Omega W^* \ , \tag{38}$$

meaning that if $\Omega = 0$ the optimal algorithm is simply the Hebb rule (W^* =constant). However, if Ω is a non zero constant, then the optimal algorithm gives an exponentially decaying weight to the newly presented example.

The generalization error asymptotic behaviour for large α_p can be obtained from equations (35) and (5):

$$e_g^{(p)} \simeq \frac{1}{\sqrt{2\pi}\,|1-\epsilon|}\ \alpha_p^{-1/2} \ . \tag{39}$$

Again it is worth pointing that no matter how much noise corrupts the teacher output, the Hebbian algorithm will always give an $\alpha_p^{-1/2}$ decay for the generalization error (unless obviously for $\epsilon = 1$ where no learning can occur).

4 Local Learning from Noiseless Data in the Tree Committee Machine

In this section we study the Tree Committee Machine (TCM). The special case in which there are only three hidden units has already been studied in [9], where it was shown that the optimal algorithm for on-line learning requires non-local information. That means that the optimal weight function for some branch perceptron depends on information of the other branches. We investigate now to what amount such non-locality is important, what is the optimal local procedure and how this problem is connected to learning in the perceptron from noisy data.

4.1 The TCM model

The K Tree Committee Machine we deal with is a set of K independent boolean perceptrons (branches) with $N_p = N_{cm}/K$ inputs units each. They do not share any input component. The notation we use is such that every N_{cm}-dimensional vector $\vec{V}$ can be thought of as K N_p-dimensional branch-vectors, *i.e.* $\vec{V} = (\mathbf{V}_1, \dots, \mathbf{V}_K)$.

We consider the case where a student network learns from a learning set provided by a teacher network, and we assume they have the same architecture. The learning set is a set of P pairs $\{(\vec{S}^\mu, \Sigma_B^\mu)\}(\mu = 1, \dots, P)$ where $\vec{S} = (\mathbf{S}_1, \dots, \mathbf{S}_K)$ with $S_{kj} = \pm 1$, $k = 1, \dots, K$, $j = 1, \dots, N_p$ and Σ_B is the teacher output (see eq. (41) below). The synaptic weights of the teacher are denoted by $\vec{B} = (\mathbf{B}_1, \dots, \mathbf{B}_K)$.

Given an input branch vector $\mathbf{S}_k$ each branch perceptron $\mathbf{B}_k$ of the teacher net gives a partial output

$$\sigma_{Bk} = \mathrm{sign}(b_k) \ , \qquad b_k \equiv \mathbf{B}_k . \mathbf{S}_k / B_k \ . \tag{40}$$

The normalization $\sum_{j=1}^{N_p} B_{kj}^2 \equiv B_k^2 = 1$ can be imposed without loss of generality.

The set $\{\sigma_{Bk}\}$ (the so called "internal representation" of $\vec{S}$ in the teacher net) is unaccessible for the student net. The only quantity it can access (besides the input vector $\vec{S}$) is the teacher's final output, which is made up from the internal representation:

$$\Sigma_B = \mathrm{sign}(\mathcal{B}) \ , \qquad \mathcal{B} \equiv \sum_{k=1}^{K} \sigma_{Bk} \tag{41}$$

The student net is defined by a vector of real connections $\vec{J}$ and upon presentation of an input vector $\vec{S}$, it gives an output

$$\Sigma_J = \mathrm{sign}(\mathcal{H}) \ , \qquad \mathcal{H} \equiv \sum_{k=1}^{K} \sigma_{Jk} \ , \tag{42}$$

where

$$\sigma_{Jk} = \mathrm{sign}(h_k) \ , \qquad h_k \equiv \mathbf{J}_k . \mathbf{S}_k / J_k \ . \tag{43}$$

As in the perceptron case, the aim of training the student TCM is to obtain $\vec{J}$, using information from the learning set, such that $P(\Sigma_J = \Sigma_B)$, upon the presentation of a random input vector uncorrelated with the learning set, is maximized. Mato and Parga [16] have calculated the generalization error $e_g^{(cm)}$ for a TCM with K hidden units as a function of the overlaps $\rho_k \equiv \mathbf{J}_k . \mathbf{B}_k / J_k B_k$, where $J_k = |\mathbf{J}_k|$. In the situation where the K overlaps have the same value, *i.e.* $\rho_k = \rho \ , \ k = 1, \dots, K$, the generalization error is given [16] by

$$e_g^{(cm)}(\rho) = \frac{1}{2} - \frac{1}{2\pi^2} \sum_{\substack{n=0 \\ (\text{even})}}^{K-1} \begin{pmatrix} K \\ n \end{pmatrix} \left[B\left(\frac{K-n}{2}, \frac{n+1}{2} \right) \right]^2 (1 - 2e_g^{(p)})^{K-n} \ , \tag{44}$$

where $e_g^{(p)}$ is the generalization error that each student branch perceptron would present with respect to the corresponding teacher branch perceptron, and B is Euler's Beta function. In the $K \to \infty$ limit, it can be shown that

$$e_g^{(cm)}(\{\rho_k\}) \overset{(\rho_k=\rho)}{=} \frac{1}{\pi} \arccos\left[\frac{2}{\pi}\arcsin(\rho)\right] . \tag{45}$$

Since we are interested in maximizing the rate at which the generalization error decreases with the presentation of examples, what eq. (44) suggests is to maximize $d\rho_k/d\alpha_{cm}$ for $k = 1, \ldots, K$, where now α_{cm} stands for the number of presented examples per machine connection

$$\alpha_{cm} = \frac{\mu}{N_{cm}} = \frac{\alpha_p}{K} . \tag{46}$$

Maximizing $de_g^{(cm)}/d\alpha_{cm}$ implies the maximization of $de_g^{(p)}/d\alpha_p$, turning the TCM on-line learning problem into a simple perceptron one. This property is due to the fact that the receptive fields are non-overlapping. But on-line learning with the simple perceptron requires knowledge of the teacher perceptron output, which in this case belongs to the unaccessible internal representation of a given example. The central issue about the equivalence between on-line learning in the committee machine and in perceptrons with noisy data is that for random distributions of examples the final machine output Σ_B can be viewed as a corrupted version of the unaccessible σ_{Bk}, which is necessary for learning at each student branch perceptron $\mathbf{J}_k$. One can then use the results of section 3 if the effective "noise level" ϵ is known. Clearly it depends on the size of the machine and one expects $\epsilon(K)$ to be a monotonously increasing function of K.

4.2 Local learning in TCMs

The equivalence property sketched above is discussed below in a more formal way. A learning algorithm in the kth TCM branch is said to be local when, besides the teacher output Σ_B, it makes use only of a variable $\mathcal{F}_k$ which does not depend on the fields $\{h_l\}_{l\neq k}$ of the other branches. $\mathcal{F}_k$ is more generally defined by some conditional probability distribution

$$P(\mathcal{F}_k \mid \{h_l\}, \{b_l\}) \overset{\text{local}}{\to} P(\mathcal{F}_k \mid h_k) \tag{47}$$

and may contain only partial information about h_k. We also define a variable $\mathcal{G}_k$ through its conditional probability distribution

$$P(\mathcal{G}_k \mid \{h_l\}, \{b_l\}) \overset{\text{local}}{\to} P(\mathcal{G}_k \mid h_k) . \tag{48}$$

$\mathcal{G}_k$ is a random variable whose informational value about the receptive field h_k is complementary to that of $\mathcal{F}_k$. For example, the optimal algorithm uses all the information contained in the local field, $\mathcal{F} = h$ and $\mathcal{G} = 1$; the Hebb algorithm has $\mathcal{F} = 1$ and $\mathcal{G} = h$; and Rosenblatt perceptron algorithm has $\mathcal{F} = \sigma_J$ and $\mathcal{G} =\mid h \mid$.

For a given set $\{\mathcal{F}_k, \mathcal{G}_k\}$, the probability distribution $P\left(b_k, \mathcal{G}_k, \Sigma_B, \mathcal{F}_k\right)$ is the most relevant quantity to be calculated. From it, the determination of an optimized algorithm and the calculation of the corresponding performance are straightforward procedures. Starting from the gaussian distribution P_0 given by eq. (78), its calculation follows as

$$\begin{aligned} P\left(b_k, \mathcal{G}_k, \Sigma_B, \mathcal{F}_k\right) &= P_0(b_k) \int \left(\prod_{l \neq k} P_0(b_l, h_l)\ db_l\ dh_l \right) P_0(h_k \mid b_k)\ dh_k \\ &\quad \times \delta\left(\Sigma_B, \mathrm{sign}\left[\sum_{l=1}^{K} \mathrm{sign}(b_l) \right] \right)\ P(\mathcal{F}_k, \mathcal{G}_k \mid h_k)\ . \end{aligned} \tag{49}$$

Integrals over $\{h_l\}_{l \neq k}$ disappear and the resulting expression can be factorized in a simple way:

$$\begin{aligned} P\left(b_k, \mathcal{G}_k, \Sigma_B, \mathcal{F}_k\right) &= P_0(b_k) \int \left(\prod_{l \neq k} P_0(b_l) db_l \right) \delta\left(\Sigma_B, \mathrm{sign}\left[\sum_{l=1}^{K} \mathrm{sign}(b_l) \right] \right) \\ &\quad \times \int P_0(h_k \mid b_k)\ dh_k\ P(\mathcal{F}_k, \mathcal{G}_k \mid h_k) \\ &= P_0(b_k)\ P(\Sigma_B \mid b_k)\ P(\mathcal{F}_k, \mathcal{G}_k \mid b_k)\ . \end{aligned} \tag{50}$$

Similarly, in the perceptron learning from noisy data described in the previous section the probability distribution $P\left(b, \mathcal{G}, \xi, \mathcal{F}\right)$ is the relevant quantity. Since $P(\xi \mid b, \mathcal{F}, \mathcal{G}) = P(\xi \mid b)$, it follows that

$$P\left(b, \mathcal{G}, \xi, \mathcal{F}\right) = P_0(b)\ P(\xi \mid b)\ P(\mathcal{F}, \mathcal{G} \mid b)\ . \tag{51}$$

So, the probability distributions for both problems factorize in a very similar manner. It remains to show that $P(\Sigma_B \mid b_k)$ has the same structure as $P(\xi \mid b)$. We can write

$$P(\xi \mid b) = P(\xi = -\mathrm{sign}(b)) + \left[1 - 2P(\xi = -\mathrm{sign}(b))\right] \Theta\left(\xi\, \mathrm{sign}(b)\right)\ , \tag{52}$$

where $P(\xi = -\mathrm{sign}(b)) = \epsilon/2$. From eq. (50), we see that for uniformly distributed examples (in which case $P(\sigma_{Bk} = \pm 1) = 1/2$), we can write

$$P(\Sigma_B \mid b_k) = P(\Sigma_B = -\mathrm{sign}(b_k)) + \left[1 - 2P(\Sigma_B = -\mathrm{sign}(b_k))\right] \Theta\left(\Sigma_B\, \mathrm{sign}(b_k)\right)\ , \tag{53}$$

where $P(\Sigma_B = -\mathrm{sign}(b_k))$ is also a constant which depends on K. Therefore, for a TCM with K hidden units there is an "effective noise level" $\epsilon(K)$ such that $P(\Sigma_B \mid b_k, K){=}P(\xi \mid b, \epsilon(K))$. The study of a local algorithm in the TCM defined by some modulation function $F_k(\Sigma_B, \mathcal{F}_k)$ is *equivalent* to the study of the same algorithm in a perceptron with examples corrupted by some noise level $\epsilon(K)$.

For this equivalence to hold it is fundamental that the optimized algorithm be fully *local*, *i.e.* the optimal weight function for a given branch perceptron k does not make use information from the other branch students internal fields $\{h_l\}_{l \neq k}$. These internal fields are correlated with the teacher branch perceptrons fields $\{b_l\}_{l \neq k}$ (see eq. (78)), and thus non-local rules allows to infer (in some sense) what the internal representation of the teacher net is. This detailed information about the relation between Σ_B and $\{b_l\}$ has no equivalent in the simpler noisy perceptron situation.

4.3 The equivalent noise level $\epsilon(K)$

The quantity in the TCM problem which corresponds to ξ in the simple perceptron problem is Σ_B. To obtain the equivalent noise level ϵ we must then calculate $P_K(\sigma_{Bk} = \Sigma_B) \equiv P(\sigma_{Bk} = \Sigma_B|K)$, which we now proceed to do. Defining

$$\tilde{\mathcal{B}}_k \equiv \sum_{l \neq k} \sigma_{Bl} \ , \tag{54}$$

which is the contribution to $\mathcal{B}$ of all the hidden units except the kth one, and recalling that the sum has $K-1$ terms (and thus is an even number), it is easy to see that

$$\begin{array}{lcl} \sigma_{Bk}\,\tilde{\mathcal{B}}_k > 0 & \Rightarrow & \Sigma_B = \sigma_{Bk} \\ \sigma_{Bk}\,\tilde{\mathcal{B}}_k = 0 & \Rightarrow & \Sigma_B = \sigma_{Bk} \\ \sigma_{Bk}\,\tilde{\mathcal{B}}_k < 0 & \Rightarrow & \Sigma_B = -\sigma_{Bk} \end{array} \tag{55}$$

so that

$$P_K(\sigma_{Bk} \neq \Sigma_B) = P_K(\sigma_{Bk}\,\tilde{\mathcal{B}}_k < 0) \ . \tag{56}$$

In the uniform distribution of examples studied here, hidden units are set to ± 1 with equal probabilities, *i.e.* $P(\sigma_{Bk}) = 1/2$, so that $P_K(\sigma_{Bk}\tilde{\mathcal{B}}_k < 0) = P_K(\tilde{\mathcal{B}}_k < 0)$. In this case, a simple counting leads to

$$P_K(\tilde{\mathcal{B}}_k) = \left(\frac{1}{2}\right)^{K-1} \frac{(K-1)!}{\left(\frac{K-1+\tilde{\mathcal{B}}_k}{2}\right)!\left(\frac{K-1-\tilde{\mathcal{B}}_k}{2}\right)!} \ . \tag{57}$$

As expected, $P_K(\tilde{\mathcal{B}}_k) = P_K(-\tilde{\mathcal{B}}_k)$, so that

$$P_K(\tilde{\mathcal{B}}_k > 0) = P_K(\tilde{\mathcal{B}}_k < 0) = \frac{1}{2}\left[1 - P_K(\tilde{\mathcal{B}}_k = 0)\right] \ . \tag{58}$$

Finally,

$$P_K(\sigma_{Bk} \neq \Sigma_B) = \frac{1}{2}\left[1 - D(K)\right] \ , \tag{59}$$

where

$$D(K) \equiv P_K(\tilde{\mathcal{B}}_k = 0) = \left(\frac{1}{2}\right)^{K-1} \frac{(K-1)!}{\left[\left(\frac{K-1}{2}\right)!\right]^2} \ . \tag{60}$$

is the probability that the k-th branch decides the machine output. Since the noise level ϵ was defined in section 3 as twice the probability that the original output σ_{Bk} is flipped, the final expression of the "equivalent noise level" is:

$$\epsilon(K) = 1 - D(K) = 1 - \left(\frac{1}{2}\right)^{K-1} \frac{(K-1)!}{\left[\left(\frac{K-1}{2}\right)!\right]^2} \ . \tag{61}$$

Thus, the optimal local on-line learning for a TCM has been solved for any number of branches K. The evolution of the branches ovelaps is calculated from eq. (23) using the effective noise level $\epsilon(K)$. This overlap gives the branch error $e_g^{(p)}(\alpha_p)$ which must be inserted in eq. (44) for obtaining the learning curve $e_g^{(cm)}(\alpha_{cm})$. Observe that in this process the change $\alpha_p = K\alpha_{cm}$ should be done.

K	ϵ	$I(\epsilon)$
1	0	2.264
3	1/2=0.5	0.375
5	5/8=0.625	0.205
7	11/16=0.6875	0.141
9	$93/128 \simeq 0.727$	0.108
11	0.754	0.087
13	0.774	0.073
15	0.790	0.063
17	0.803	0.055
19	0.814	0.049
21	0.824	0.044

Table 1: Noise level ϵ and the corresponding asymptotic integral $I(\epsilon)$ as functions of K.

4.4 Asymptotic behaviour

For finite K, $I(\epsilon(K))$ is non-zero (see eq. (26)), implying a α_{cm}^{-1} decay for the generalization error per branch perceptron $e_g^{(p)}$, according to eqs. (28) and (46). According to eq. (44), the asymptotic generalization error of the machine behaves like

$$e_g^{(cm)} \simeq C(K)\, e_g^{(p)} , \tag{62}$$

i.e. a constant (which grows with K) times the asymptotic generalization error per branch perceptron (*e.g.* for $K=3$ we have $e_g^{(cm)} \simeq 3/2\, e_g^{(p)}$, asymptotically). This happens because the sum on eq. (44) has a finite number of terms, so that the term of order $[e_g^{(p)}]^1$ dominates. Then, the optimal local algorithm always leads to a α_{cm}^{-1} decay for the generalization error, as long as K remains finite. In Table 1 the noise level ϵ and its corresponding integral $I(\epsilon)$ are shown for several values of K.

A subtle point concerns the meaning of $K \to \infty$ and the onset of the asymptotical behaviour. Due to the non uniform convergence of eq. (44) as $K \to \infty$, at $\rho \to 1$ there is a crossover in behaviour of $e_g^{(cm)}(\rho)$. For finite but large K two regimes can be detected. For intermediate values of α_{cm}, the error decays as $\alpha_{cm}^{-1/2}$, while for sufficiently large α_{cm} it crosses over to α_{cm}^{-1}. The crossover occurs for increasingly larger values of α_{cm} as $K \to \infty$ ($\alpha_{cross} \propto \sqrt{K}$). Thus, in the limit of infinite K the α_{cm}^{-1} regime cannot be reached (see Fig. 3).

The asymptotic behaviour of $D(K)$, for large K, is

$$D(K) = 1 - \epsilon(K) \simeq \sqrt{\frac{2}{\pi}} \frac{1}{\sqrt{K}} , \tag{63}$$

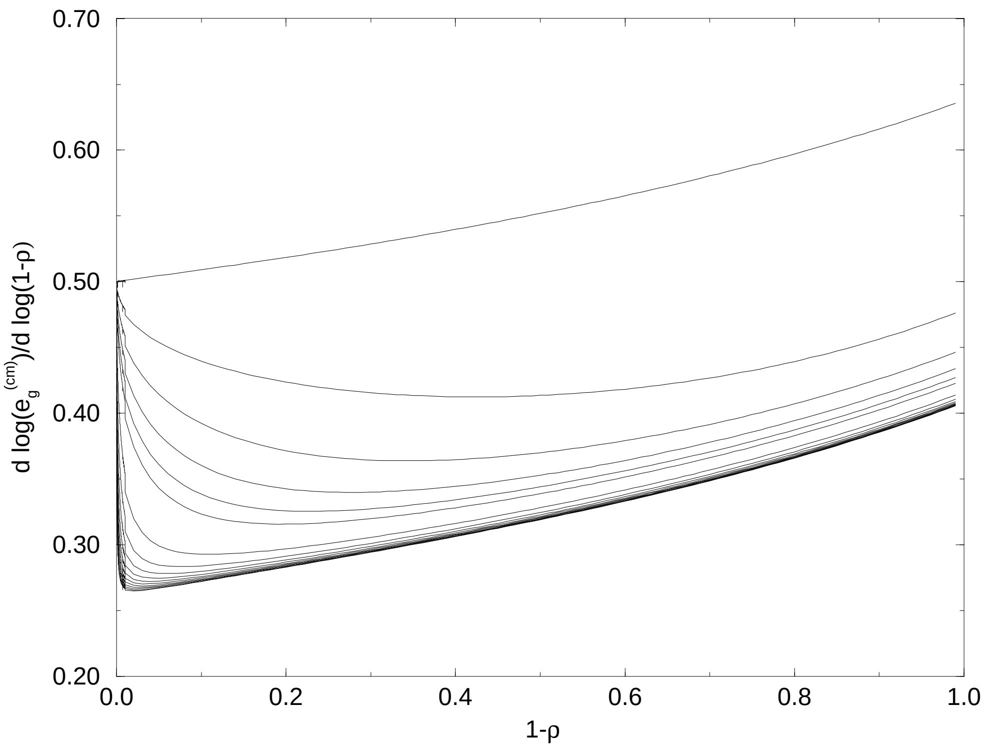

Figure 3: Non uniform convergence of $e_g^{(cm)}(\rho)$ as $K \to \infty$. For large but finite K, $d\log(e_g^{(cm)})/d\log(1-\rho)$ approaches $1/4$ for small values of $1-\rho$. Yet, for ρ sufficiently close to 1, the expression converges to $1/2$. From top to bottom, $K = 1, 3, 5, 7, 9, 11, 21, 31, 41, 51, 61, 71, 81, 91, 101, 111, 121$.

so that $\epsilon(K) \to 1$ in this limit, as expected. From eq. (29) it follows that

$$I(\epsilon\,(K)) \stackrel{K\to\infty}{\simeq} \frac{2\sqrt{2}}{\pi}\frac{1}{K}\,, \tag{64}$$

so that the $e_g^{(p)}$ asymptotic decay can be obtained from eqs. (28) and (64):

$$e_g^{(p)} \simeq \frac{\pi}{\sqrt{2}}\frac{K}{\alpha_p} = \frac{\pi}{\sqrt{2}}\,\alpha_{cm}^{-1}\,. \tag{65}$$

The above equation shows that if the number of presented examples is proportional to the number of connections in each branch perceptron (that is, finite α_p), no learning can occur for infinitely many hidden units. The proper scaling in this case happens to be the number of presented examples per machine connection (α_{cm}). Since the noise level is close to unity, the modulation function prevents the occurrence of major synaptic changes in a broader region of the Δ axis. This could be interpreted as if each branch perceptron learned in the average only once every K examples. Eq. (45) and (65) lead then to an asymptotic ($\alpha_{cm} \to \infty$) generalization error

$$e_g^{(cm)} \simeq \frac{2}{\pi}\sqrt{e_g^{(p)}} \simeq \frac{2^{3/4}}{\sqrt{\pi}}\, \alpha_{cm}^{-1/2} \, . \tag{66}$$

Simulations were performed to check the equivalence described, and results are shown in Figure 4. While using the optimal weight (22), the unaccessible overlap ρ was replaced by the norm J as measured during run time, and the agreement between theoretical calculations and simulations show that flutuactions on J are absorbed without further consequences. Note that while symbols refer to simulations on TCMs (with K from 1 to 11), solid lines represent the numerical integration of eq. (23) for the corresponding $\epsilon(K)$, confirming the equivalence property.

A similar analysis can be applied to the simple Hebb learning. From eqs. (39) and (62) it follows that for finite K the machine generalization error has a $\alpha_{cm}^{-1/2}$ asymptotic decay. For large K, one finds again that the number of presented examples must be proportional to the number of input units of the committee machine for learning to occur. Using result (63) it follows that

$$e_g^{(p)} \simeq \frac{1}{2}\sqrt{\frac{K}{\alpha_p}} = \frac{1}{2}\, \alpha_{cm}^{-1/2} \, . \tag{67}$$

In the $K \to \infty$ limit, $e_g^{(cm)}$ is given by eq. (45). Near $\rho = 1$, result (67) can be used, leading to

$$e_g^{(cm)} \simeq \frac{\sqrt{2}}{\pi}\, \alpha_{cm}^{-1/4} \, . \tag{68}$$

Simulations for the simple Hebb algorithm on the TCM can be seen in Fig. (5).

4.5 Relation to standard algorithms

It is very interesting to compare the optimal algorithm, which has been theoretically determined from a variational argument, with the heuristical algorithms proposed for these machines (TCMs with boolean units), such as Mitchison and Durbin Least Action procedure [17] and its variants.

The modulation function F^{opt} produces major synaptic changes in the hidden units which have negative but small stabilities Δ. The Least Action algorithm follows a procedure which crudely resembles the optimal function: only the synapses of the unit with lowest (in modulo) negative stability are updated. The decreasing of F^{opt} along the learning process produced by the prefactor λ is also approximated by the conventional algorithms by using a decreasing learning step. The optimal local algorithm gives then a theoretical rationale for the success of these algorithms.

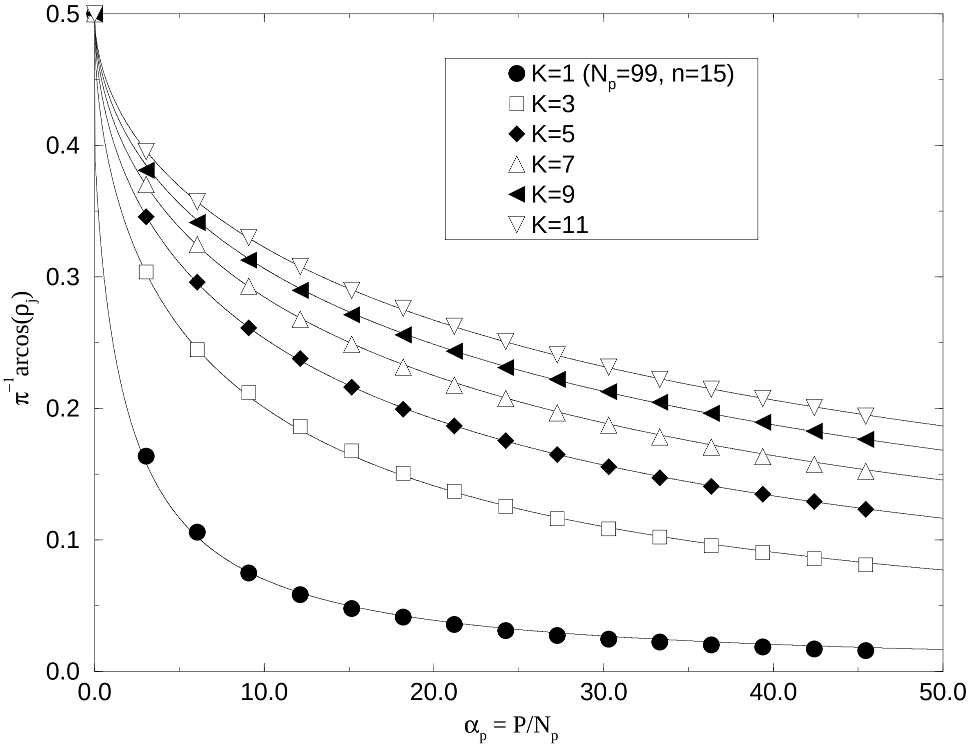

Figure 4: Average of $e_g^{(p)}$ taken over K branches against α_p, for the optimal algorithm. N_p is the number of units per perceptron and n is the number of runs, for each K. See text for details.

Of course, the optimal modulation function has a much richer structure. The heuristic algorithms can be thought as crude approximations of the local optimal prescription. Higher performances will be obtained by those algorithms which approximates better the shape and behaviour of F^{opt}. The optimal nonlocal algorithm is even more sofisticate: in some circunstances the lowest stability branch is not the one that gets the largest correction [9].

5 Learning by Queries

Kinzel and Ruján [2] pointed out that by appropriately choosing the new examples, with a distribution $P(h) = \delta(h)$, the generalization error of a single layer perceptron could be reduced from an asymptotic decay of $0.399/\alpha^{1/2}$ to $0.199/\alpha^{1/2}$. In [6] it was shown that if the modulation function was permitted to evolve along the learning process, a vast quali-

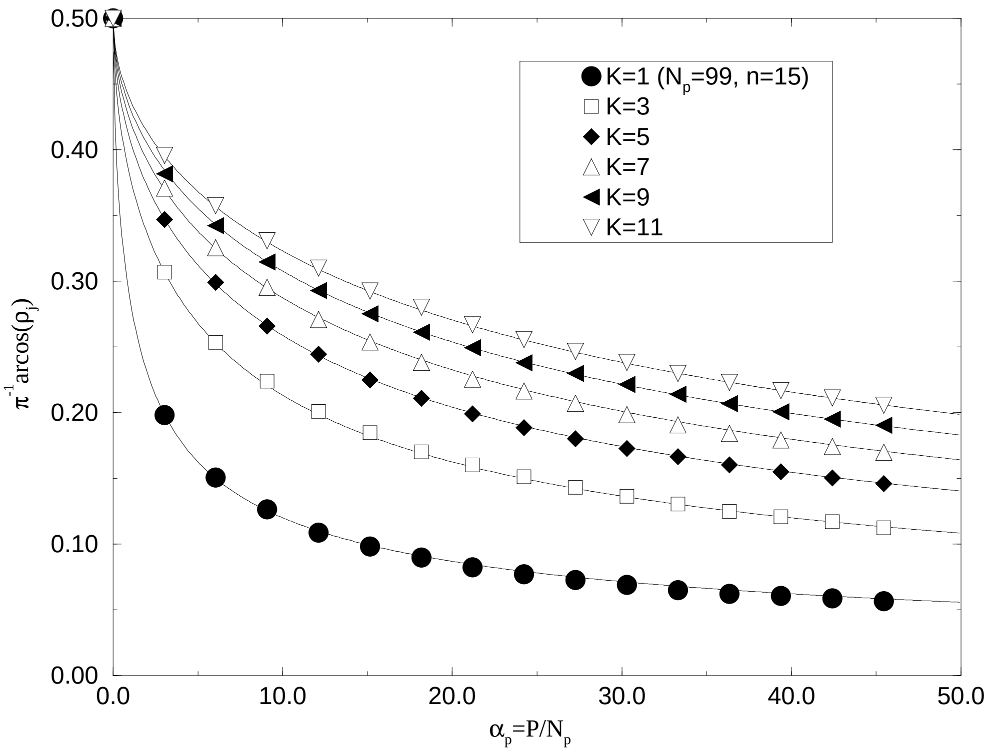

Figure 5: Average of $e_g^{(p)}$ taken over K branches against α_p, for the Hebbian algorithm. N_p is the number of units per perceptron and n is the number of runs, for each K.

tative improvement could be achieved, with a resulting exponential decay. We extend this results now in two directions. We first show that the exponential decay of the perceptron generalization error is preserved even in the presence of noise and secondly that this results in an exponentially decaying error for the TCM, according to the equivalence discussed in the preceding section. For large K this decay is independent of K.

In obtaining eq. (23) we made explicit use of the distribution $P_0(h)$. For an arbitrary distribution function $P(h)$ (for which the conditional distribution function $P(b \mid h)$ is still $P_0(b \mid h)$) the evolution of ρ is given by

$$\frac{d\rho}{d\alpha_p} = \frac{(1-\rho^2)}{\rho}\frac{(1-\epsilon)^2}{4\pi}\int dh\, P(h)\left[\frac{e^{-h^2/\lambda^2}}{\epsilon/2+(1-\epsilon)H(-h/\lambda)} + \frac{e^{-h^2/\lambda^2}}{\epsilon/2+(1-\epsilon)H(h/\lambda)}\right]. \qquad (69)$$

Defining the term between square brackets as $f(h)$, $d\rho/d\alpha_p$ is maximized if the distribution of h is chosen to be $P(h) = \delta(h - h_0)$, where h_0 is the maximum of f. The solution of $df(h_0)/dh_0 = 0$ is easily seen to be $h_0 = 0$, leading then to the optimized distribution $P_{opt}(h) = \delta(h)$, a generalization of the results of reference [6]. Inserting $P_{opt}(h)$ into eq. (69), one obtains

$$\frac{d\rho}{d\alpha_p} = \frac{(1-\epsilon)^2}{\pi} \frac{(1-\rho^2)}{\rho} , \tag{70}$$

which for the initial condition $\rho(0) = 0$ has the solution

$$\rho = \left[1 - \exp\left(-2(1-\epsilon)^2 \alpha_p/\pi\right)\right]^{1/2} . \tag{71}$$

When $\alpha_p \to \infty$, the corresponding branches error are described by

$$e_g^{(p)} \simeq \frac{1}{\pi} \exp\left(-(1-\epsilon)^2 \alpha_p/\pi\right) . \tag{72}$$

This means that for finite K the committee machine error decays exponentially with α_{cm}, with a K-dependent prefactor. For large K, results (63) and (72) can be used, leading to the branch error

$$e_g^{(p)} \simeq \frac{1}{\pi} \exp\left(-2\alpha_{cm}/\pi^2\right) , \tag{73}$$

which is independent of K. In the $K \to \infty$ limit we have

$$e_g^{(cm)} \simeq \frac{2}{\pi^{3/2}} \exp\left(-\alpha_{cm}/\pi^2\right) . \tag{74}$$

To implement this strategy it is necessary to select input vectors $\mathbf{S}$ orthogonal to $\mathbf{J}$, which may be a costly search. An observation not made previously in the literature is that the exponential decay can be achieved by a less rigorous choice of $\mathbf{S}$, say, vectors which have $h = c\lambda$. In this case we obtain from equation (69),

$$\frac{d\rho}{d\alpha_p} = \frac{(1-\epsilon)^2}{\pi} \frac{(1-\rho^2)}{\rho} \mathcal{I}(c) , \tag{75}$$

$$\mathcal{I}(c) = \frac{e^{-c^2/2}}{4} \left[\frac{1}{\epsilon/2 + (1-\epsilon)H(-c)} + \frac{1}{\epsilon/2 + (1-\epsilon)H(c)}\right] , \tag{76}$$

which also leads to an exponential decay for the generalization error with a prefactor $\mathcal{I}(c) < 1$ in front of α_{cm} in the expressions for the errors.

6 Summary and Conclusions

The variational approach to the determination of optimal algorithms has already given interesting results for machines with diverse architectures: linear perceptrons [18] and boolean perceptrons [6], parity machines [19] and committee machines with boolean units [9]. Although this approach has been mainly used in conjunction with on-line learning, it can also be applied to off-line learning [20]. Interestingly, the optimal F^{opt} has the same form in both scenarios.

The important message from this approach is that learning algorithms can be *theoretically derived* instead of being proposed only from *ad hoc* and heuristical considerations. All the standard algorithms suggested in the literature for these machines can be viewed as more or less reliable approximations of the optimal modulation function F^{opt} determined for each learning situation.

The study of increasingly richer architectures should be undertaken based on the hope that some properties will remain valid, being more a reflection of generic learning behaviour rather than lucky outcomes due to the restrict set of architectures examined. One general result which has emerged is the form of the optimal cost function, $E_{opt} = -\lambda^2 P(\xi \mid \{h_k\})$, which is valid for all learning situations and all machines studied so far. The connection with Bayesian inference ideas need to receive a more deep exploration.

On-line optimal learning in perceptrons provides a clear instance of a truly adaptive method for changing environments, having the following interesting properties:

- the training is incremental (no retraining over past examples is required);
- the training phase is not decoupled from the performance phase;
- changes in the enviromment (rule) can be detected and tracked automatically;
- it is robust to noise in the examples and 'outliers';
- it does not present overfitting (on-line learning is a radical "early stopping" strategy);
- learning curves can be calculated analytically;
- it is computationally cheap and efficient.
- it has biological appeal;

Usual measures of performance take into account the dependence of the generalization error as a function of the number of examples used in the process. Even by these standards, optimized on-line learning competes well with iterated, computationally expensive off-line methods. If CPU time to attain a given error is taken as a measure of performance then, depending on the price of new examples, on-line learning can really beat other learning schemes.

We have presented lower bounds on generalization errors. That these might be attained hinges on the knowledge of the noise level. We have shown that the determination of noise level in the perceptron translates into architecture determination in the committee machine. The *a priori* knowledge of an architecture amounts to the determination of confidence levels on the data used for training and thus any restriction in the determination of one should apply to the other. Therefore methods to determine the noise level ϵ, as the one developed by Biehl *et al.* [13], are important since they may also suggest methods for estimation of the teacher architecture complexity in other problems.

Although local algorithms for TCMs works well and have the nice equivalence with algorithms for the noisy perceptron problem, the study of *non-local* algorithms for the general K TCM does not loose its importance. On the contrary, the performance difference between local ($e_g^{(cm)} \simeq 2.67/\alpha_{cm}$) and non-local ($e_g^{(cm)} \simeq 0.88/\alpha_{cm}$) optimal on-line algorithms in the $K=3$ case signals that non-locality plays an important role during learning. Moreover, taking into account that information provided by non-local algorithms becomes more important for larger values of K (when compared to local algorithms), the motivation for searching for the non-local optimal performances certainly increases.

Acknowledgements: OK would like to thank Rosângela Loschi for useful discussions about Bayesian inference. The work of MC was supported by Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), while OK and NC received support from Conselho Nacional de Pesquisa Científica e Tecnológica (CNPq).

Note: After the main results of this work had been obtained we received a preprint from Biehl *et al.* [13] which contains some overlapping results about learning in the perceptron with noise.

A Training energy for the optimal algorithm in the case of random examples

The dynamic evolution may be thought of as a stochastic gradient descent minimization of some cost function $H(\mu)$ defined in the space of normalized vectors $\hat{\mathbf{J}} \equiv \mathbf{J}/J$

$$\hat{J}_i(\mu) = \hat{J}_i(\mu-1) - \frac{1}{N_p}\frac{\partial H}{\partial \hat{J}_i}\,, \tag{77}$$

where $\hat{J}_i \equiv J_i/J$ is the ith component of the unit vector $\hat{\mathbf{J}}$. The total energy $H = V + E$ has two parts, the potential $V^\mu = \frac{1}{2}\Omega(\mu)\hat{\mathbf{J}}\cdot\hat{\mathbf{J}}$ controlling the synaptic vector length and the *training energy* E^μ which depends only on the example μ. We list below the most studied algorithms:

- *Hebb:* $E_H^\mu = -\Delta_\mu\,;\ F_H(\mu) = \xi^\mu$;
- *Adaline:* $E_A^\mu = \frac{1}{2}(\kappa-\Delta_\mu)^2\,;\ F_A(\mu) = (\kappa-\Delta_\mu)\,\xi^\mu$;
- *Perceptron:* $E_P^\mu = (\kappa-\Delta_\mu)\,\Theta(\kappa-\Delta_\mu)\,;\ F_P(\mu) = \Theta(\kappa-\Delta_\mu)\,\xi^\mu$;
- *Relaxation:* $E_R^\mu = (\kappa-\Delta_\mu)^2\,\Theta(\kappa-\Delta_\mu)\,;\ F_R(\mu) = (\kappa-\Delta_\mu)\,\Theta(\kappa-\Delta_\mu)\,\xi^\mu$.

where κ is a constant (or perhaps a function of α_p) often called the *Gardner parameter.*

Consider distributions of input vectors such that the fields b and h (as defined in 2.1) are gaussian correlated variables with zero mean[1] and unit variance:

$$\begin{aligned} P_0(b,h) &= P_0(h) \times P_0(b \mid h) \\ &= \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{h^2}{2}\right) \times \frac{1}{\sqrt{2\pi(1-\rho^2)}} \exp\left(-\frac{(b-\rho h)^2}{2(1-\rho^2)}\right) . \end{aligned} \tag{78}$$

Starting from this probability distribution, we wish to show that the optimal weight function (11) has a corresponding training energy

$$E_{opt} = -\lambda^2 \ln P(\xi \mid h) \,, \qquad \lambda \equiv \frac{\sqrt{1-\rho^2}}{\rho} \,. \tag{79}$$

The deduction that follows is strictly valid for the random distribution of examples (78) above, though we have reasons to believe that eq. (79) is a general prescription (see below).

According to eqs. (8) and (77), the weight function is related to the $\hat{\mathbf{J}}$ gradient of the energy function. Starting then from eq. (79), it follows that

$$\begin{aligned} \frac{\partial E_{opt}}{\partial \hat{J}_i} &= -\frac{\lambda^2}{P(\xi \mid h)} \frac{\partial}{\partial \hat{J}_i} P(\xi \mid h) \\ &= -\frac{\lambda^2}{P(\xi \mid h)} \frac{\partial}{\partial \hat{J}_i} \int db \; P(\xi \mid b) P(b \mid h) \,, \end{aligned} \tag{80}$$

where we have used that $P(\xi \mid b,h) = P(\xi \mid b)$, so that the only dependence on $\hat{J}_i$ lies on the student receptive field h of $P(b \mid h)$. Only now must we introduce the specific examples distribution (78). For $P(b \mid h) = P_0(b \mid h)$ we have

$$\frac{\partial}{\partial \hat{J}_i} P_0(b \mid h) = \frac{S_i}{\lambda^2} \left(\frac{b}{\rho} - h\right) P_0(b \mid h) \,, \tag{81}$$

so that, returning to eq. (80),

$$\begin{aligned} \frac{\partial E_{opt}}{\partial \hat{J}_i} &= -\frac{\lambda^2}{P(\xi \mid h)} \int db \; P(\xi \mid b) P_0(b \mid h) \frac{S_i}{\lambda^2} \left(\frac{b}{\rho} - h\right) \\ &= -S_i \int db \; P(b \mid \xi, h) \left(\frac{b}{\rho} - h\right) \\ &= -S_i \frac{F^{opt}}{J} \,. \end{aligned} \tag{82}$$

Since in this derivation we have not specified what is ξ, it holds both for Boolean as graded response perceptrons. For example, ξ may be the proper field b, or a noisy version of it, so that we are dealing with the linear perceptron problem [18].

---

[1] if the means are non zero, we can work with the normalized fields $\hat{b} = b - \langle b \rangle$ and $\hat{h} = h - \langle h \rangle$.

Result (82) enables us then to give an alternative prescription for obtaining the optimal weight function F^{opt}. It is much easier to calculate the derivative

$$F^{opt} = -S_i J \frac{\partial E_{opt}}{\partial \hat{J}_i} = S_i \lambda^2 J \frac{\partial}{\partial \hat{J}_i} \ln P(\xi \mid h) \tag{83}$$

than to perform the integration over b prescribed by eq. (11).

It is worth pointing that prescriptions (11) and (83) holds independently of our particular choice of noise (3). For example, noise could be added to the teacher field b instead, and both prescriptions would still be valid.

Now we show that the Bayesian prescription of maximizing the probability of occurrence of the hypothesis given the data leads to E_{opt}. Note that

$$\begin{aligned} P(\xi \mid \hat{\mathbf{J}}, \mathbf{S}) &= \int dh\ P(\xi \mid h) P(h \mid \hat{\mathbf{J}}, \mathbf{S}) \\ &= \int dh\ P(\xi \mid h) \delta(h - \hat{\mathbf{J}}.\mathbf{S}) \\ &= P(\xi \mid \hat{\mathbf{J}}.\mathbf{S})\ , \end{aligned} \tag{84}$$

so that

$$\begin{aligned} \ln P(\hat{\mathbf{J}} \mid \xi, \mathbf{S}) &= \ln P(\xi \mid \hat{\mathbf{J}}, \mathbf{S}) + \ln P(\hat{\mathbf{J}}) - \ln P(\xi) \\ &= \ln P(\xi \mid h) + \text{constant}\ , \end{aligned} \tag{85}$$

once the *a priori* distribution for $\hat{\mathbf{J}}$ is uniform. This may not be the case in more involved situations with a non-uniform hypothesis space.

It is important to note that the relevant quantity is $\hat{\mathbf{J}}$, not h. The Bayesian prescription in our case is to maximize the function $\ln P(\hat{\mathbf{J}} \mid \xi, \mathbf{S})$, *not* $\ln P(h \mid \xi)$. The later would lead to an extra term, say $\ln P_0(h) = -h^2/2$, which is *not* a constant. The difference between $P(\hat{\mathbf{J}} \mid \xi, \mathbf{S})$ and $P(h \mid \xi)$ lies in the fact that the sum $h = \sum_i y_i$ of a large number of variables ($y_i = J_i S_i$) with *uniform* priors has a *non-uniform* (Gaussian) prior (central limit theorem).

This relationship with Bayesian inference leads us to conjecture that the prescription given by eq. (83) continues to be valid for other network architectures and learning situations where $P(\mathbf{J})$ is uniform. Of course, if $P(\mathbf{B})$ is known to be non-uniform, this may lead us to use an adequate prior $P(\mathbf{J})$ so that improved performances may be attainable.

References

[1] Watkin, T. L. H., Rau, A., Biehl, M., "The statistical mechanics of learning a rule", *Rev. Mod. Phys.* **65** 2 (1993)

[2] Kinzel, W., Ruján, P., "Improving a Network Generalization Ability by Selecting Examples", *Europhys. Lett.* **13** 473 (1990)

[3] Kinouchi O., Caticha N., "Biased learning in boolean perceptrons", *Physica* **185A** 411 (1992)

[4] Biehl, M., Schwarze, H., "Learning by on-line gradient descent", *J.Phys. A: Math. Gen.* **28** 643 (1995)

[5] Kinouchi, O., Caticha, N., "Lower bounds on generalization errors for drifting rules", *J. Phys. A: Math. Gen.* **26** 6161 (1993)

[6] Kinouchi O., Caticha N., "Optimal generalization in perceptrons", *J.Phys. A: Math. Gen.* **25** 6243 (1992)

[7] Opper M., Haussler D., "Generalization performance of Bayes optimal classification algorithm for learning a perceptron", *Phys. Rev. Lett.* **66** 2677 (1991)

[8] Kabashima, Y., "Perfect loss of generalization due to noise in K=2 parity machines", *J.Phys. A: Math. Gen.* **27** 1917 (1994)

[9] Copelli, M., Caticha, N., "On-line learning in the committee machine", *J.Phys. A: Math. Gen.* **28** 1615-1625 (1995)

[10] Saad, D., Solla, S., "Exact solution for on-line learning in multilayer neural networks", *Phys. Rev. Lett.* **74** 4337-4340 (1995)

[11] Saad, D., Solla, S., "On-line learning in soft committee machines", to appear in *Phys. Rev. E* (1995)

[12] Riegler P. and Biehl, M., "On-line backpropagation in two-layered neural networks", Preprint WUE-ITP-95-018 Würzburg University (1995)

[13] Biehl, M., Riegler, P., Stechert, M., "Learning from noisy data: an exactly solvable model", Preprint WUE-ITP-95-001 Würzburg University (1995)

[14] Vallet, F., "The Hebb rule for learning linearly separable boolean functions: learning and generalization", *Europhys. Lett.* **8** (8) 747 (1989)

[15] Izquierdo, I., Netto, C. A. and Quillfeldt, J. A., "Memory modulation" *in* "Neural networks and spin glasses", Theumann, W., Köberle, R., Eds. (World Scientific, 1989)

[16] Mato, G., Parga, N., "Generalization properties of multilayered neural networks", *J. Phys. A: Math. Gen.* **25** 5047 (1992)

[17] Mitchison, G. J. and Durbin, R. M., "Bounds on the learning capacity of some multi-layer networks", *Biol. Cybern.* **60** 345 (1989).

[18] Kinouchi, O., Caticha N., "On-line versus off-line learning in the linear perceptron: a comparative study", to appear in *Phys. Rev. E.*

[19] Simonetti, R. and Caticha, N. “Optimal on-line learning in the parity machine” preprint (1995)

[20] Kinouchi, O., Caticha N., “A learning algorithm which gives the Bayes generalization limit for perceptrons”, Preprint IFUSP/P-1156 Universidade de São Paulo (1995)